WHAT IS CLAIMED IS:

1. A method for using a redundant word in a register file in place of a faulty word in the register file comprising the following steps:
   fetching an original instruction from a first storage;
   mapping a logical address having a logical addressing space in the original instruction to a physical address having a physical addressing space, the physical addressing space being larger than the logical addressing space;
   if the logical address points to the faulty word in the register file, making the physical address the address of the redundant word, the redundant word being in the physical addressing space, but the redundant word not being in the logical addressing space; and
   if the logical address does not point to the faulty word in the register file, using the logical address as the physical address.

2. The method of claim 1, further comprising the step of storing the physical address into a predecoded instruction.

3. The method of claim 2, wherein the original instruction has a plurality of checking bits, the step of storing the physical address into the predecoded instruction further comprising the step of using a bit in the predecoded instruction corresponding to a checking bit in the original instruction to accommodate the larger physical addressing space of the physical address, allowing the predecoded instruction to have an equal number of bits as the original instruction.

4. The method of claim 2, further comprising the step of storing the predecoded instruction into a second storage.

5. The method of claim 2, further comprising the step of:

fetching the predecoded instruction from the second storage; and

using the physical address in the predecoded instruction by a decode to access a word in the register file, where a redundant word can be accessed by the decode.

6. The method of claim 1, wherein the step of using the logical address as the physical address comprises the steps of:

using the logical address as a group of lower order bits in the physical address; and

placing one or more binary "0" bits as a group of higher order bits in the physical address;

wherein the physical address consists of the lower order bits in the physical address and the higher order bits in the physical address.

7. The method of claim 1 further comprising the steps of:

identifying a faulty word in the register file;

determining if a fault in the faulty word is a soft fail or a hard fail;

if the fault is hard fail, changing the mapping of the logical address to the physical address of the redundant word; and

if the fault is a soft fail, not changing the mapping of the logical address to the physical address of the redundant word.

8. A computer system comprising:

a register file having a number of words within an addressing space of a logical address, and a redundant word not addressable by the logical address;

a predecode unit that maps the logical address to a physical address, the physical address having a larger address space than the logical address;

a register addressing portion capable of accessing any word within the addressing space of the physical address, without use of an address compare; and

a diagnostic unit capable of identifying a faulty word in the register file and controlling the predecode unit to map the logical address to a physical address responsive to the identity of the faulty word.

9. The computer system of claim 8, wherein the logical address is stored in an original instruction and the physical address is stored in a predecoded instruction.

10. The computer system of claim 9, wherein the original instruction contains a plurality of logical addresses and the predecoded instruction contains a plurality of physical addresses.

11. The computer system of claim 9 further comprising:
a first storage containing the original instruction, the predecode unit coupled to the first storage and capable of reading the original instruction from the first storage; and
a second storage coupled to the predecode unit and capable of receiving a predecoded instruction from the predecode unit.

12. The computer system of claim 11 wherein the predecode unit is capable of reading a plurality of original instructions from the first storage simultaneously and the second storage is capable of receiving a plurality of predecoded instructions simultaneously from the predecode unit.

13. The computer system of claim 11, wherein the first storage is a second level cache and the second storage is a first level cache.

14. The computer system of claim 10, wherein the first storage is a third level cache and the second storage is a second level cache.

15. The computer system of claim 8, the diagnostic unit further comprising:

a diagnostic routine capable of determining if the faulty word has a soft fail or a hard fail, and only controlling the predecode unit to map the logical address to a physical address responsive to the identity of the faulty word if the faulty word has a hard fail.

16. The computer system of claim 8, the predecode unit further comprising:

a remap table having a fault address column and a replacement address column and a number of rows, each row containing a fault address that is the logical address of a faulty word and a replacement physical address that points to a redundant word in the register file; and

a remap unit, coupled to the logical address, to the remap table, and to the predecoded instruction, that compares the logical address to all of the fault addresses in the remap table, and, if a match occurs, stores the corresponding replacement address in the physical address in the predecoded instruction, but storing the logical address in the least significant bits of the physical address in the predecoded instruction, with one or more binary "0"s in one or more most significant bits of the physical address, if no matches are found.

17. The computer system of claim 16, wherein the remap table is included in the remap unit.

18. The computer system of claim 9, further comprising:

a checking bit in the original instruction; and

an extended bit in the predecoded instruction in a bit corresponding to the checking bit in the original instruction used to accommodate the larger addressing space of the physical address in the predecoded instruction, allowing the predecoded instruction to be the same number of bits as the original instruction.